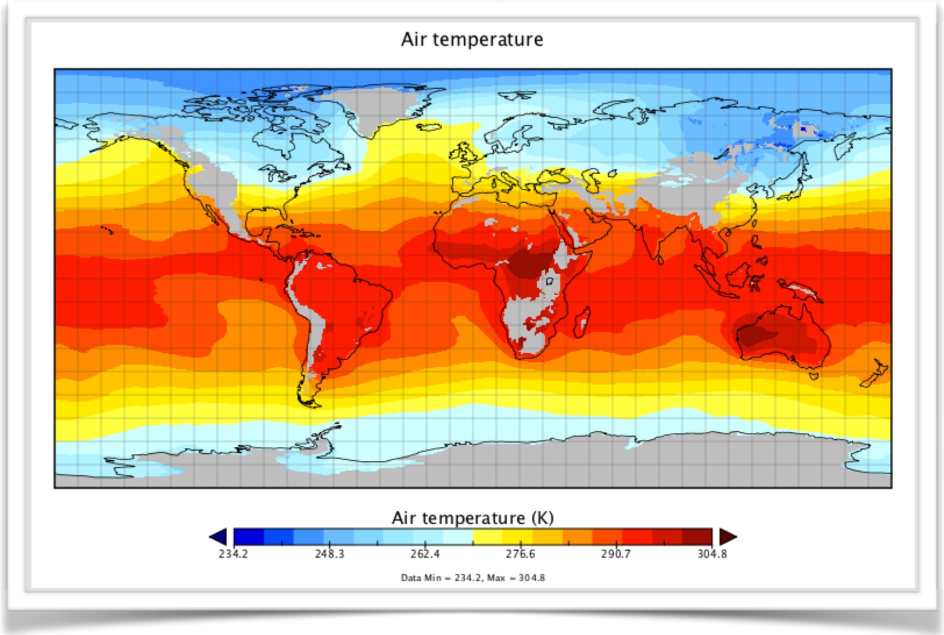
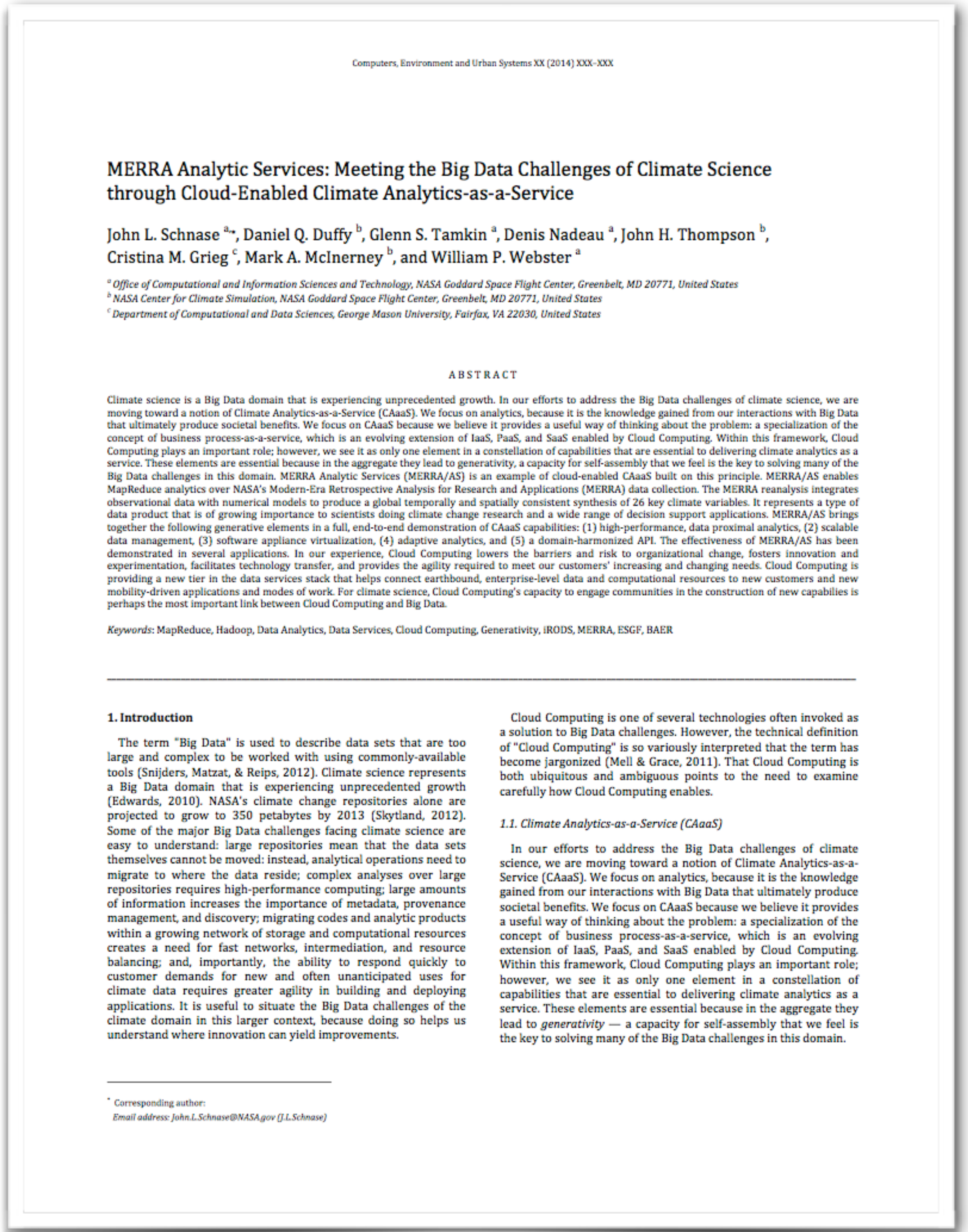
MERRA Analytic Services (MERRA/AS)

John Schnase, Glenn Tamkin, Dan Duffy, and Mark McInerney

NASA Goddard Space Flight Center

March 20, 2014





Computers, Environment and Urban Systems XX (2014) XXX–XXX

MERRA Analytic Services: Meeting the Big Data Challenges of Climate Science through Cloud-Enabled Climate Analytics-as-a-Service

John L. Schnase [a,*], Daniel Q. Duffy [b], Glenn S. Tamkin [a], Denis Nadeau [a], John H. Thompson [b], Cristina M. Grieg [c], Mark A. McInerney [b], and William P. Webster [a]

[a] *Office of Computational and Information Sciences and Technology, NASA Goddard Space Flight Center, Greenbelt, MD 20771, United States*
[b] *NASA Center for Climate Simulation, NASA Goddard Space Flight Center, Greenbelt, MD 20771, United States*
[c] *Department of Computational and Data Sciences, George Mason University, Fairfax, VA 22030, United States*

ABSTRACT

Climate science is a Big Data domain that is experiencing unprecedented growth. In our efforts to address the Big Data challenges of climate science, we are moving toward a notion of Climate Analytics-as-a-Service (CAaaS). We focus on analytics, because it is the knowledge gained from our interactions with Big Data that ultimately produce societal benefits. We focus on CAaaS because we believe it provides a useful way of thinking about the problem: a specialization of the concept of business process-as-a-service, which is an evolving extension of IaaS, PaaS, and SaaS enabled by Cloud Computing. Within this framework, Cloud Computing plays an important role; however, we see it as only one element in a constellation of capabilities that are essential to delivering climate analytics as a service. These elements are essential because in the aggregate they lead to generativity, a capacity for self-assembly that we feel is the key to solving many of the Big Data challenges in this domain. MERRA Analytic Services (MERRA/AS) is an example of cloud-enabled CAaaS built on this principle. MERRA/AS enables MapReduce analytics over NASA's Modern-Era Retrospective Analysis for Research and Applications (MERRA) data collection. The MERRA reanalysis integrates observational data with numerical models to produce a global temporally and spatially consistent synthesis of 26 key climate variables. It represents a type of data product that is of growing importance to scientists doing climate change research and a wide range of decision support applications. MERRA/AS brings together the following generative elements in a full, end-to-end demonstration of CAaaS capabilities: (1) high-performance, data proximal analytics, (2) scalable data management, (3) software appliance virtualization, (4) adaptive analytics, and (5) a domain-harmonized API. The effectiveness of MERRA/AS has been demonstrated in several applications. In our experience, Cloud Computing lowers the barriers and risk to organizational change, fosters innovation and experimentation, facilitates technology transfer, and provides the agility required to meet our customers' increasing and changing needs. Cloud Computing is providing a new tier in the data services stack that helps connect earthbound, enterprise-level data and computational resources to new customers and new mobility-driven applications and modes of work. For climate science, Cloud Computing's capacity to engage communities in the construction of new capabilies is perhaps the most important link between Cloud Computing and Big Data.

Keywords: MapReduce, Hadoop, Data Analytics, Data Services, Cloud Computing, Generativity, iRODS, MERRA, ESGF, BAER

1. Introduction

The term "Big Data" is used to describe data sets that are too large and complex to be worked with using commonly-available tools (Snijders, Matzat, & Reips, 2012). Climate science represents a Big Data domain that is experiencing unprecedented growth (Edwards, 2010). NASA's climate change repositories alone are projected to grow to 350 petabytes by 2013 (Skytland, 2012). Some of the major Big Data challenges facing climate science are easy to understand: large repositories mean that the data sets themselves cannot be moved: instead, analytical operations need to migrate to where the data reside; complex analyses over large repositories requires high-performance computing; large amounts of information increases the importance of metadata, provenance management, and discovery; migrating codes and analytic products within a growing network of storage and computational resources creates a need for fast networks, intermediation, and resource balancing; and, importantly, the ability to respond quickly to customer demands for new and often unanticipated uses for climate data requires greater agility in building and deploying applications. It is useful to situate the Big Data challenges of the climate domain in this larger context, because doing so helps us understand where innovation can yield improvements.

Cloud Computing is one of several technologies often invoked as a solution to Big Data challenges. However, the technical definition of "Cloud Computing" is so variously interpreted that the term has become jargonized (Mell & Grace, 2011). That Cloud Computing is both ubiquitous and ambiguous points to the need to examine carefully how Cloud Computing enables.

1.1. Climate Analytics-as-a-Service (CAaaS)

In our efforts to address the Big Data challenges of climate science, we are moving toward a notion of Climate Analytics-as-a-Service (CAaaS). We focus on analytics, because it is the knowledge gained from our interactions with Big Data that ultimately produce societal benefits. We focus on CAaaS because we believe it provides a useful way of thinking about the problem: a specialization of the concept of business process-as-a-service, which is an evolving extension of IaaS, PaaS, and SaaS enabled by Cloud Computing. Within this framework, Cloud Computing plays an important role; however, we see it as only one element in a constellation of capabilities that are essential to delivering climate analytics as a service. These elements are essential because in the aggregate they lead to *generativity* — a capacity for self-assembly that we feel is the key to solving many of the Big Data challenges in this domain.

[*] Corresponding author:
Email address: John.L.Schnase@NASA.gov (J.L.Schnase)

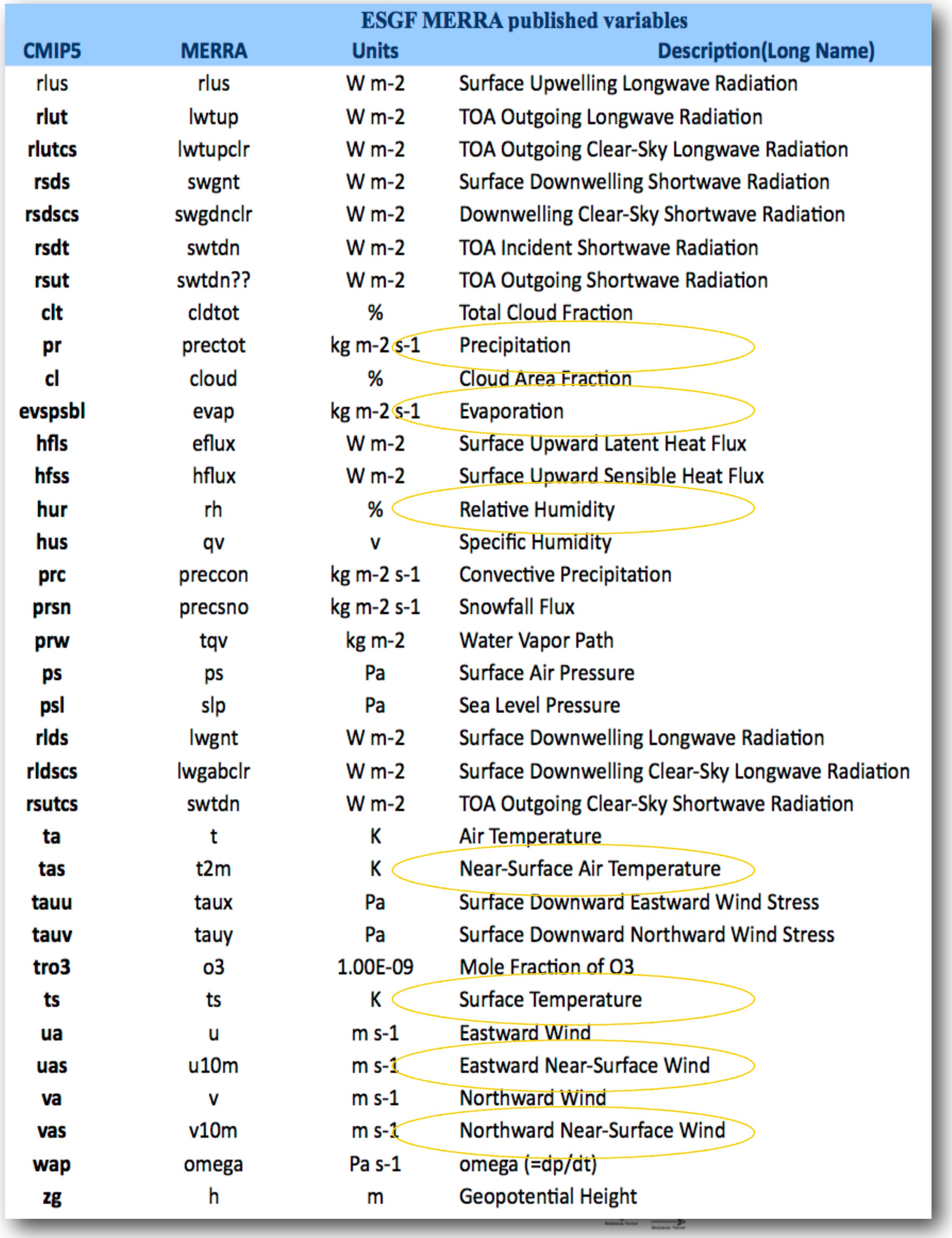
MERRA

MERRA: Modern Era-Retrospective Analysis for Research and Applications

- The MERRA reanalysis integrates observational data with numerical models to produce a global temporally and spatially consistent synthesis of 26 key climate variables.
- Spatial resolution is 1/2˚ latitude × 2/3˚ longitude × 72 vertical levels extending through the stratosphere.
- Temporal resolution is 6-hours for three-dimensional, full spatial resolution, extending from 1979-Present.
- ~ 200 TB, but MERRA II is on the way …
- Subset published through ESGF for MIP activities.

Why focus on reanalysis data?

- There is an increasing demand for reanalysis data products by a large and diverse applications community representing all of NASA’s Applied Sciences Program’s themes: disasters, ecological forecasting, health and air quality, water resources, agriculture, climate energy, oceans, and weather.
- To be useful, climate model outputs and predictions must be made easily accessible to an expanding community of consumers, including local governments, federal agencies, and private-sector customers.



ESGF MERRA published variables

CMIP5	MERRA	Units	Description(Long Name)
rlus	rlus	W m-2	Surface Upwelling Longwave Radiation
rlut	lwtup	W m-2	TOA Outgoing Longwave Radiation
rlutcs	lwtupclr	W m-2	TOA Outgoing Clear-Sky Longwave Radiation
rsds	swgnt	W m-2	Surface Downwelling Shortwave Radiation
rsdscs	swgdnclr	W m-2	Downwelling Clear-Sky Shortwave Radiation
rsdt	swtdn	W m-2	TOA Incident Shortwave Radiation
rsut	swtdn??	W m-2	TOA Outgoing Shortwave Radiation
clt	cldtot	%	Total Cloud Fraction
pr	prectot	kg m-2 s-1	Precipitation
cl	cloud	%	Cloud Area Fraction
evspsbl	evap	kg m-2 s-1	Evaporation
hfls	eflux	W m-2	Surface Upward Latent Heat Flux
hfss	hflux	W m-2	Surface Upward Sensible Heat Flux
hur	rh	%	Relative Humidity
hus	qv	v	Specific Humidity
prc	preccon	kg m-2 s-1	Convective Precipitation
prsn	precsno	kg m-2 s-1	Snowfall Flux
prw	tqv	kg m-2	Water Vapor Path
ps	ps	Pa	Surface Air Pressure
psl	slp	Pa	Sea Level Pressure
rlds	lwgnt	W m-2	Surface Downwelling Longwave Radiation
rldscs	lwgabclr	W m-2	Surface Downwelling Clear-Sky Longwave Radiation
rsutcs	swtdn	W m-2	TOA Outgoing Clear-Sky Shortwave Radiation
ta	t	K	Air Temperature
tas	t2m	K	Near-Surface Air Temperature
tauu	taux	Pa	Surface Downward Eastward Wind Stress
tauv	tauy	Pa	Surface Downward Northward Wind Stress
tro3	o3	1.00E-09	Mole Fraction of O3
ts	ts	K	Surface Temperature
ua	u	m s-1	Eastward Wind
uas	u10m	m s-1	Eastward Near-Surface Wind
va	v	m s-1	Northward Wind
vas	v10m	m s-1	Northward Near-Surface Wind
wap	omega	Pa s-1	omega (=dp/dt)
zg	h	m	Geopotential Height

MERRA Analytic Services

MERRA/AS is a cyberinfrastructure resource that ...

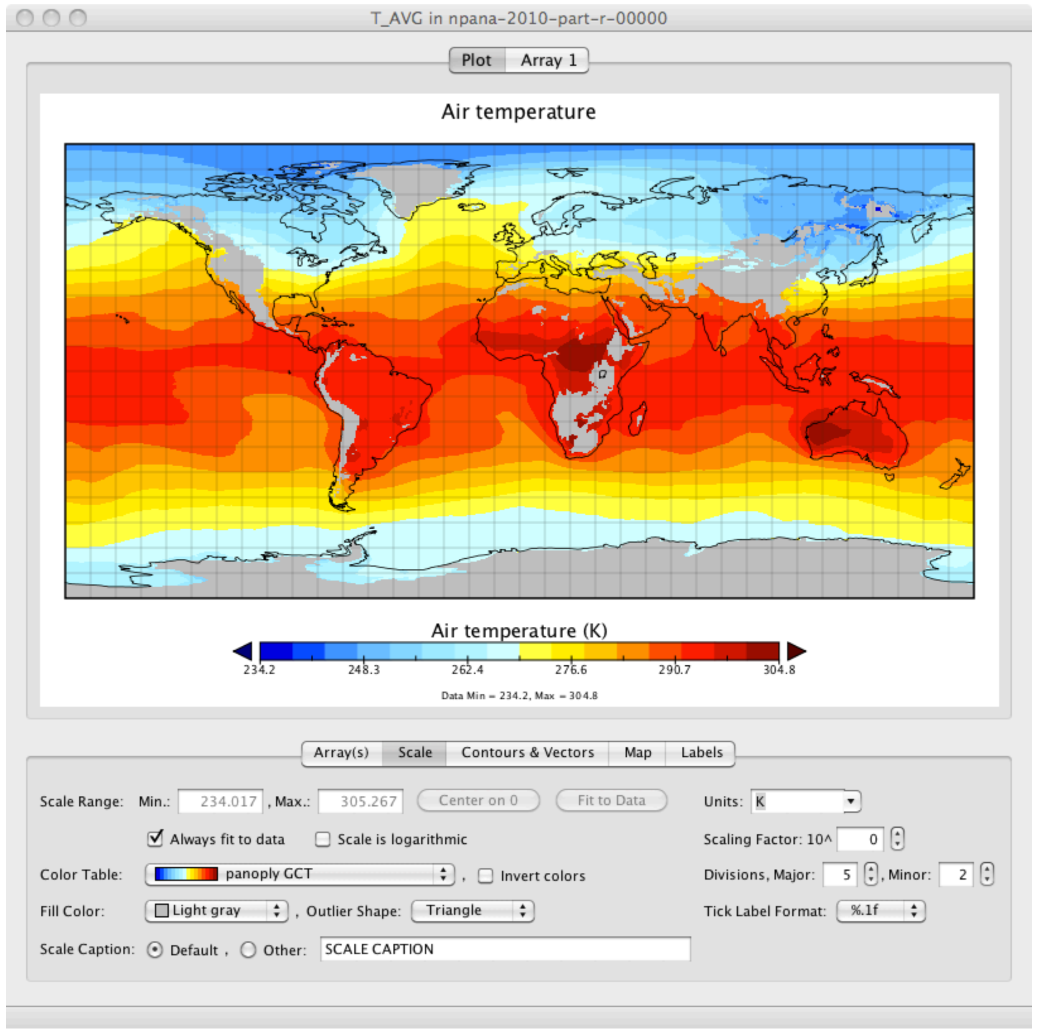
- Combines iRODS data grid and Hadoop MapReduce capabilities to serve MERRA analytic products ...

  ... or, specifically, we combines iRODS-based Climate Data Server (CDS) capabilities with Cloudera MapReduce to serve MERRA analytic products;

- Stores the MERRA reanalysis data collection in an HDFS to enable parallel, high-performance, storage-side data reductions;
- Manages storage-side <driver, mapper, reducer> code sets and realized objects for users; and
- Provides a library of commonly used spatiotemporal operations (canonical ops) that can be composed to enable higher-order analyses.

In addition, the project ...

- Makes available to the extended community iRODS/CDS "kits" that enable HDFS as a storage resource and MapReduce code hosting and utilities to externalize MERRA's metadata and sequence /de-sequence MERRA data; and
- Provides a preliminary evaluation of cost, performance, and usability of MapReduce in general and the system in particular.



Scale Range: Min.: 234.017 , Max.: 305.267

[✓] Always fit to data ☐ Scale is logarithmic

Color Table: panoply GCT , ☐ Invert colors

Fill Color: Light gray , Outlier Shape: Triangle

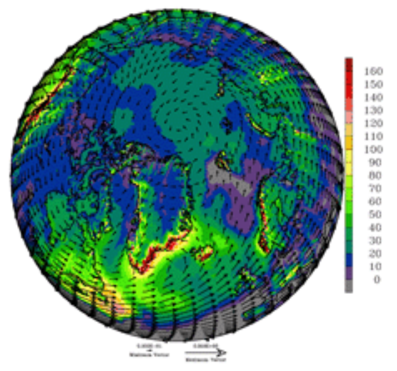
Scale Caption: [● Default] , ○ Other: SCALE CAPTION

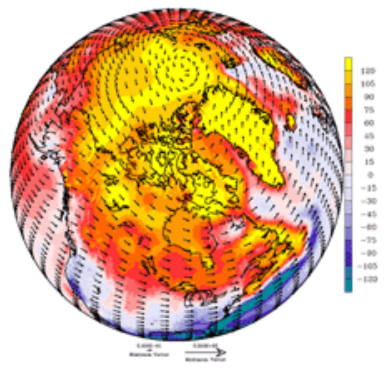
Units: K

Scaling Factor: 10^ 0

Divisions, Major: 5 , Minor: 2

Tick Label Format: %.1f

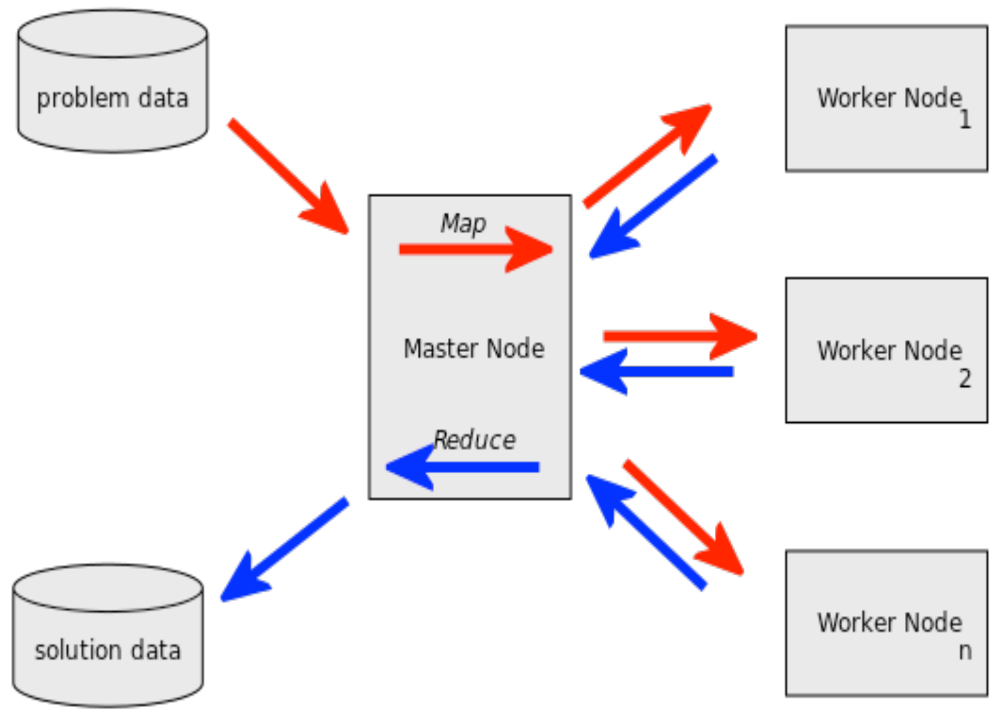




MapReduce

The Basic MapReduce Paradigm ...

- MapReduce is a framework for processing parallelizable problems across huge datasets using a large number of computers.
- Computational processing can occur on data stored either in a filesystem (unstructured) or in a database (structured).
- MapReduce can take advantage of locality of data, processing data on or near the storage assets to decrease transmission of data.
- "Map" step: The master node takes the input, divides it into smaller sub-problems, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller problem, and passes the answer back to its master node.
- "Reduce" step: The master node then collects the answers to all the sub-problems and combines them to form the output – the answer to the problem it was originally trying to solve.



Writing a parallel-executable program has proven over the years to be a very challenging task, requiring various specialized skills. MapReduce provides regular programmers the ability to produce parallel distributed programs more easily, by requiring them to write only the simpler Map() and Reduce() functions, which focus on the logic of the specific problem at hand, while the "MapReduce System" automatically takes care of marshaling the distributed servers, running the various tasks in parallel, managing all communications and data transfers between the various parts of the system, providing for redundancy and failures, and overall management of the process ...

MERRA/AS Cluster

Cluster / Node Configuration

- 36 node Dell cluster, 576 Intel 2.6 GHz SandyBridge cores, 1300 TB raw storage, 1250 GB RAM, 11.7 TF theoretical peak compute capacity.
- FDR Infiniband network with peak TCP/IP speeds >20 Gbps.

Component	Configuration
Node	Dell R720
Processor Type	Intel SandyBridge
Processor Number	E5-2670
Processor Speed	2.6 GHz
Cores per Socket	8
Number of Sockets	2
Cores per Node	16
Main Memory	32 GB
Storage	12 by 3 TB drives = 36 TB RAW
Interconnect	Mellanox MT27500 FDR IB
Operating System	CentOS 6.3
Kernel	2.6.32-279.5.1
Hadoop	0.20.2
Java	1.6.0_24

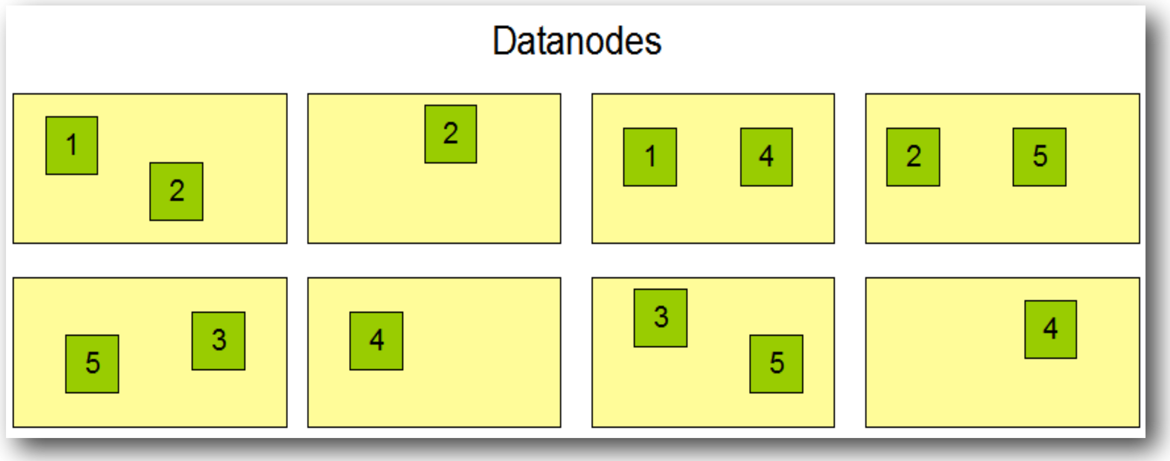
MERRA/AS Cluster

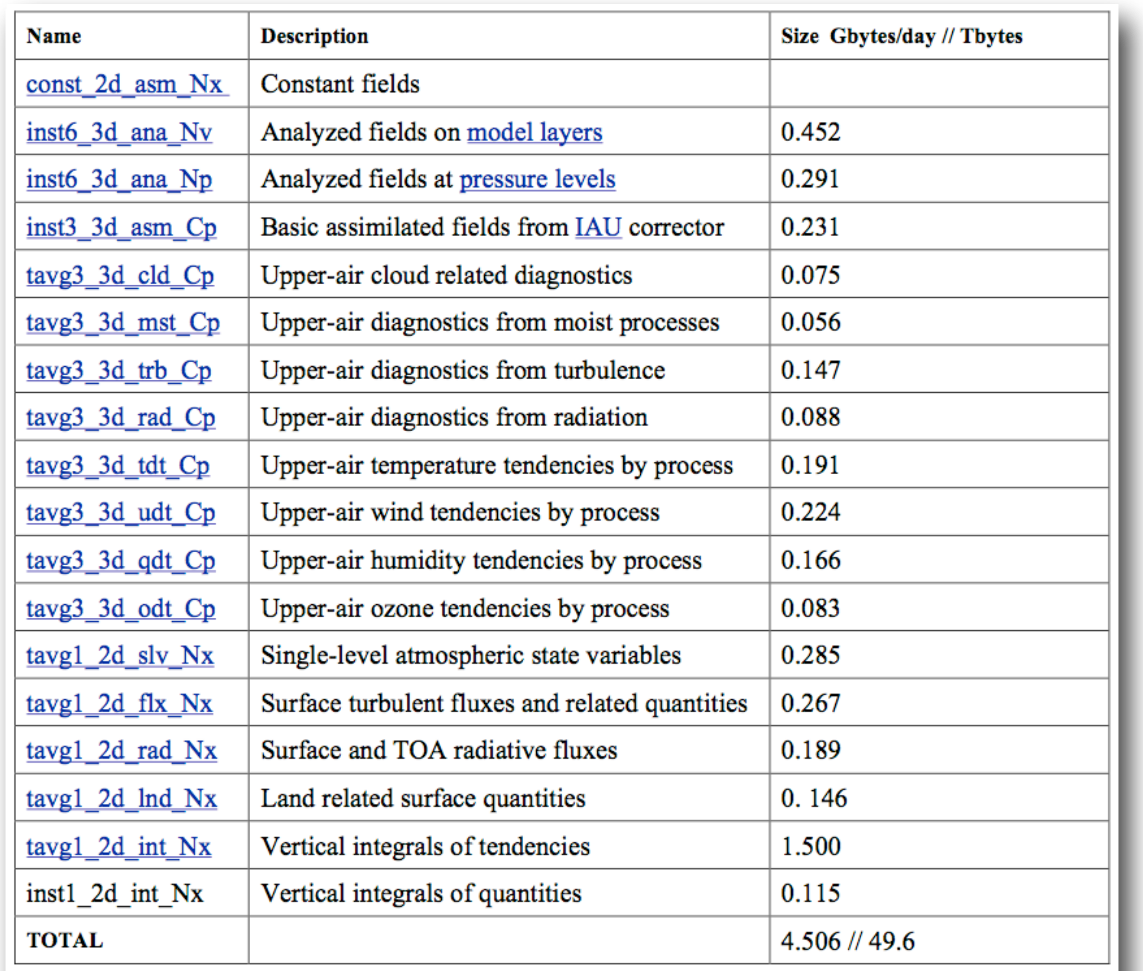
MERRA Data

- The GEOS-5 MERRA products are divided into 25 collections: 18 standard products, 7 chemistry products.
- Comprise monthly means files and daily files at six-hour intervals running from 1979 – 2012.
- Total size of the native, compressed NetCDF MERRA collection in a standard filesystem is ~80 TB.
- One file per day produced with file sizes ranging from ~20 MB to ~1.5 GB. Files obtained from the GES DISC.

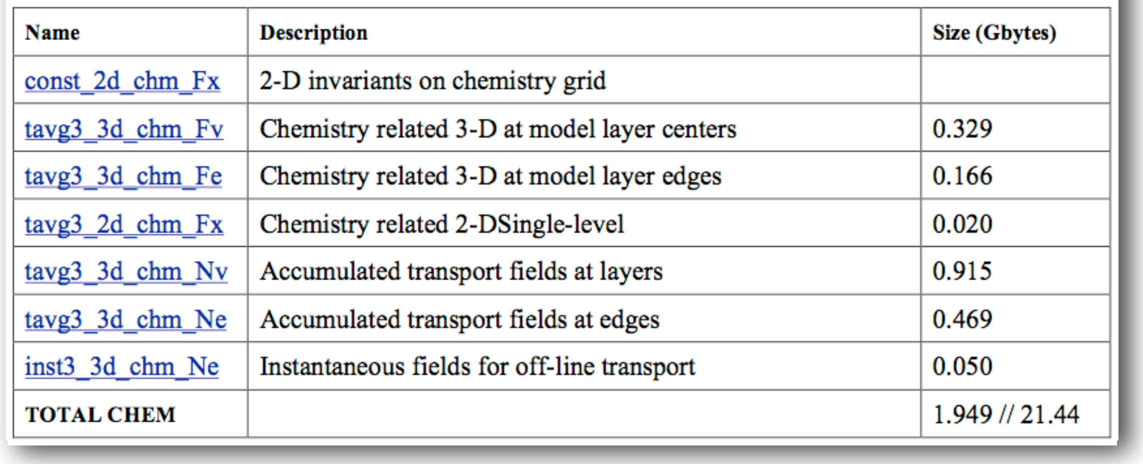
HDFS Organization

- Native MERRA files are sequenced and ingested into the Hadoop cluster in triplicated 640 MB blocks.
- Total size of the MERRA/AS HDFS repository is ~480 TB.





Name	Description	Size Gbytes/day // Tbytes
const_2d_asm_Nx	Constant fields	
inst6_3d_ana_Nv	Analyzed fields on model layers	0.452
inst6_3d_ana_Np	Analyzed fields at pressure levels	0.291
inst3_3d_asm_Cp	Basic assimilated fields from IAU corrector	0.231
tavg3_3d_cld_Cp	Upper-air cloud related diagnostics	0.075
tavg3_3d_mst_Cp	Upper-air diagnostics from moist processes	0.056
tavg3_3d_trb_Cp	Upper-air diagnostics from turbulence	0.147
tavg3_3d_rad_Cp	Upper-air diagnostics from radiation	0.088
tavg3_3d_tdt_Cp	Upper-air temperature tendencies by process	0.191
tavg3_3d_udt_Cp	Upper-air wind tendencies by process	0.224
tavg3_3d_qdt_Cp	Upper-air humidity tendencies by process	0.166
tavg3_3d_odt_Cp	Upper-air ozone tendencies by process	0.083
tavg1_2d_slv_Nx	Single-level atmospheric state variables	0.285
tavg1_2d_flx_Nx	Surface turbulent fluxes and related quantities	0.267
tavg1_2d_rad_Nx	Surface and TOA radiative fluxes	0.189
tavg1_2d_lnd_Nx	Land related surface quantities	0. 146
tavg1_2d_int_Nx	Vertical integrals of tendencies	1.500
inst1_2d_int_Nx	Vertical integrals of quantities	0.115
TOTAL		4.506 // 49.6



Name	Description	Size (Gbytes)
const_2d_chm_Fx	2-D invariants on chemistry grid	
tavg3_3d_chm_Fv	Chemistry related 3-D at model layer centers	0.329
tavg3_3d_chm_Fe	Chemistry related 3-D at model layer edges	0.166
tavg3_2d_chm_Fx	Chemistry related 2-DSingle-level	0.020
tavg3_3d_chm_Nv	Accumulated transport fields at layers	0.915
tavg3_3d_chm_Ne	Accumulated transport fields at edges	0.469
inst3_3d_chm_Ne	Instantaneous fields for off-line transport	0.050
TOTAL CHEM		1.949 // 21.44

Canonical Operations

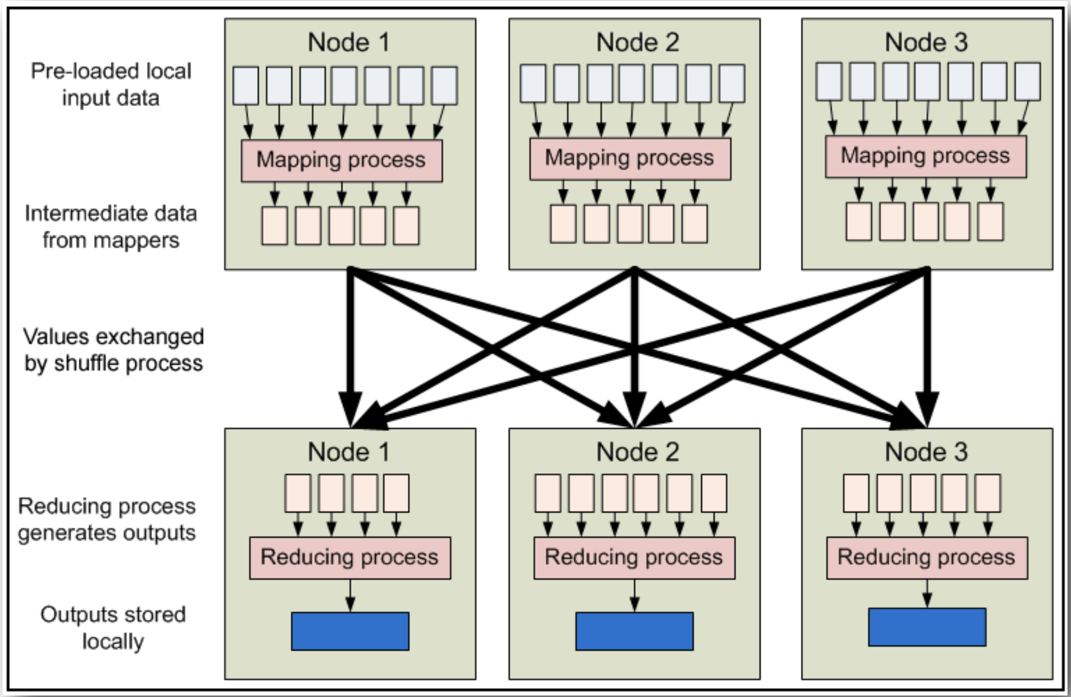
Canonical Ops Library

- We're also creating a small set of canonical near-storage, early-stage analytical operations that represent a common starting point in many analysis workflows in many domains. For example, avg, max, min, var, sum, count operations of the general form:

  result <= avg(var, (t_0,t_1), ((x_0,y_0,z_0),(x_1,y_1,z_1))),

  that return, in this example, the average of a variable when given a variable name, temporal extent, and spatial extent ...

- Averages over time, space, and elevation can be performed now for all MERRA variables.

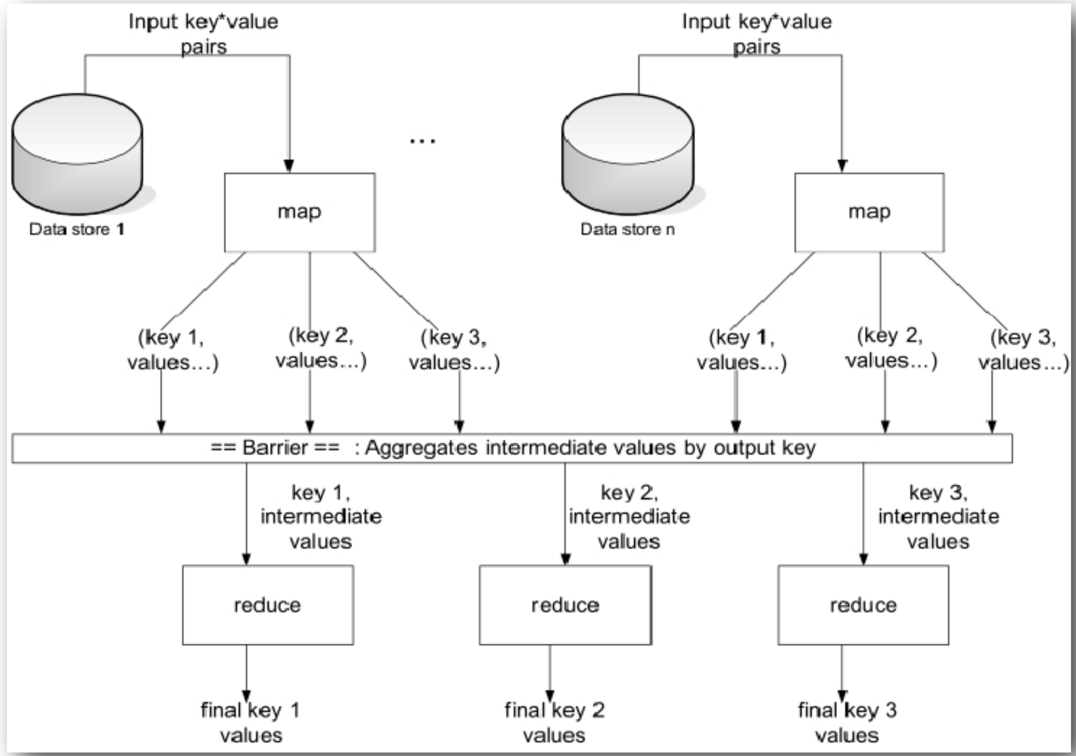

Pre-loaded local input data
Intermediate data from mappers
Values exchanged by shuffle process
Reducing process generates outputs
Outputs stored locally
Node 1
Node 2
Node 3
Mapping process
Reducing process

Rationale for canonical ops *– Data intensive analysis workflows can be thought of as directed acyclic graphs where nodes represent computations over data and arcs the systematic steps of the analysis process.*

These workflows bridge between the largely unstructured mass of archived scientific data and the highly structured, tailored, reduced, and refined analytic products that are used by individual scientists and form the basis of intellectual work in the domain.

In general, the initial steps of an analysis, those operations that first interact with a data repository, tend to be the most general, while data manipulations closer to the client tend to be the most specialized to the individual, to the domain, or to the science question under study. The amount of data being operated on also tends to be larger on the repository-side of the workflow, smaller toward the client-side end products.

This stratification can be exploited in order to optimize efficiencies along the workflow chain. MapReduce, for example, seeks to improve efficiencies of the near-archive operations that initiate workflows.

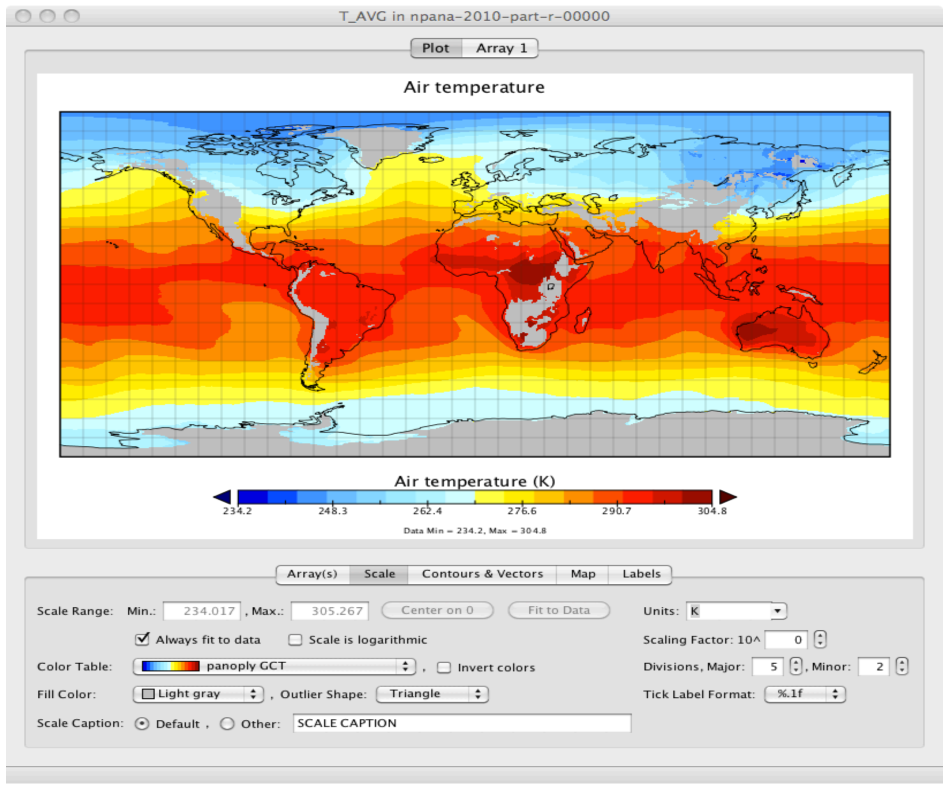


MERRA/AS Server

Code Modules

- There's a substantial code ecosystem behind the averaging operation. Lines of code per logical module is as follows:

-	Core	2617
-	Sequencer	824
-	De-sequencer	808
-	MapReduce	1138
-	Avg Op	234 (Mapper 89, Reducer 145)
	Total	**5621** <==



Scale Range: Min.: 234.017 , Max.: 305.267

✓ Always fit to data ☐ Scale is logarithmic

Color Table: panoply GCT , ☐ Invert colors

Fill Color: Light gray , Outlier Shape: Triangle

Scale Caption: ● Default , ○ Other: SCALE CAPTION

Units: K

Scaling Factor: 10^ 0

Divisions, Major: 5 , Minor: 2

Tick Label Format: %.1f

NetCDF Sequencing Common Classes — These are wrapper classes that would be used to contain the NetCDF data in a form that can be read and written to sequence files.

- *NetCDFCompositeKey* — This is the key object used by both the sequencer and the application. The key uses the field name and the date-time from the NetCDF file. This allows <key, value> pairs to be sorted by time and grouped by field.
- *NetCDFSequenceFileRecord* — This is the main data class. A record contains the main field variable, along with any other variables that were associated with this field from the NetCDF file. It also stores the essential metadata associated with the variable. It contains methods that convert NetCDF variable Java objects into this record object.
- *NetCDFSequenceFileVariable* — Contains the data and metadata/attributes from a NetCDF variable. This class also contains multiple convenience routines for accessing the data.
- *NetCDFSequenceFileAttribute* — Attributes are metadata associated with a variable. The sequence variable class uses this class to store variable attributes.

NetCDF Sequence Application Classes — These are the main sequencing and de-sequencing application classes. These rely on the common sequence classes to translate NetCDF files to sequence files and vise versa.

- *NetCDFSequenceFileGenerator* — This class opens, reads, and translates input NetCDF files into Hadoop Map files (sequence files with indexing). It uses the NetCDFCompositeKey class to construct keys, and classes from the common sequence directory to construct values. The values are serialized using a library called Kryo that packs Java objects into byte arrays.
- *SequenceToNetCDF* — Output from a MapReduce program is itself a sequence file. This class converts sequence file results into a NetCDF file to return to the MERRA/AS server.

Ops Library Classes — As part of the MERRA/AS delivery, we will begin a library of these basic operations. These canonical ops will provide a template for users as they begin their exploration of MapReduce analytics and will be useful in their own right as steps in larger analyses.

- *NetCDFAveragerMapper* — This code basically compares the current <key, value> pair to the criteria for what fields to process (in this case a simple average). Any field that does not match is rejected. Fields that are accepted are passed on to the reducer.
- *NetCDFAveragerReducer* — Upon receiving all the <key, value> pairs from the mapper, this routine goes through the grouped and ordered <key, value> pairs and performs the averaging operation based on the specified time period. When a time period has been processed, a new <key, value> pair is created and written out to disk.

Climate Data Services API

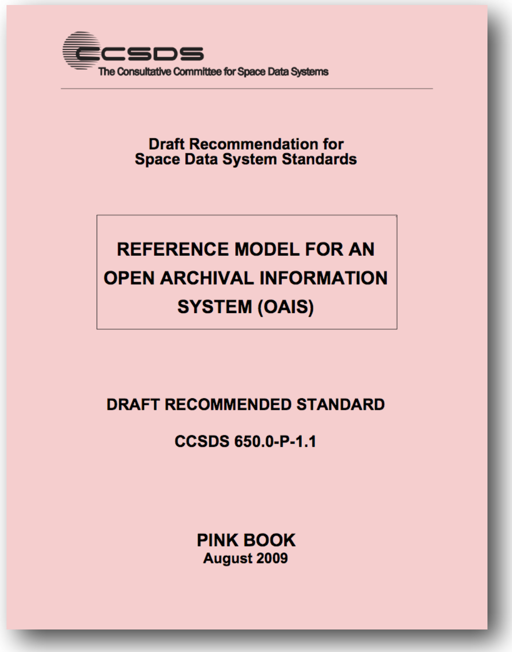
Toward Climate Analytics-as-a-Service (CAaaS) ...

- The MERRA/AS project is the starting point for development of the NASA Climate Data Services (CDS) Application Programming Interface (API).
- MapReduce functionality will be the first service delivered through the CDS API.
- The CDS API leverages the Open Archival Information System (OAIS) reference model.

CDS Reference Model

- Logical specification that presents a single abstract data and analytic services model to calling applications.
- Can be implemented using various technologies; in all cases, however, actions are based on the following six primitives:

- ***Ingest*** – Submit/register a Submission Information Package.
- ***Query*** – Retrieve data from a pre-determined service request (synchronous).
- ***Order*** – Request data from a pre-determined service request (asynchronous).
- ***Download*** – Retrieve a Dissemination Information Package.
- ***Status*** – Track progress of service activity.
- ***Execute*** – Initiate a service-definable extension.


CCSDS
The Consultative Committee for Space Data Systems

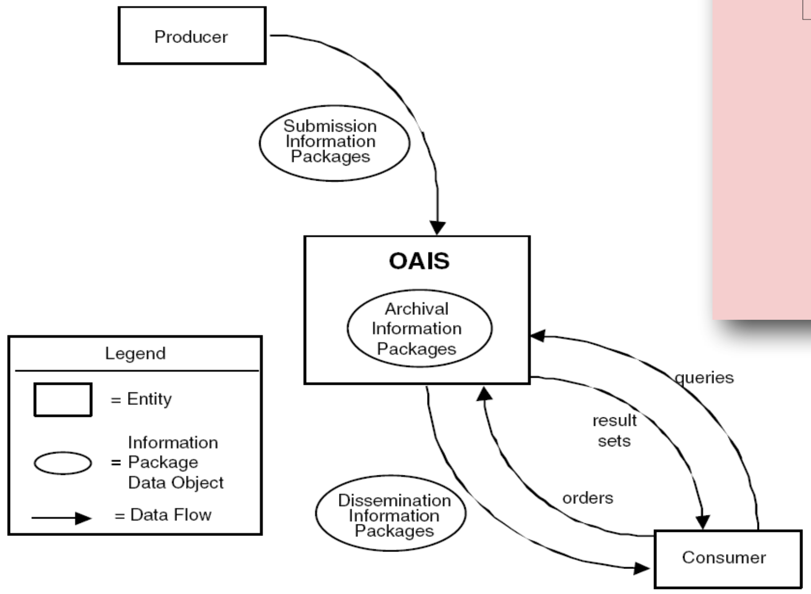
Draft Recommendation for
Space Data System Standards

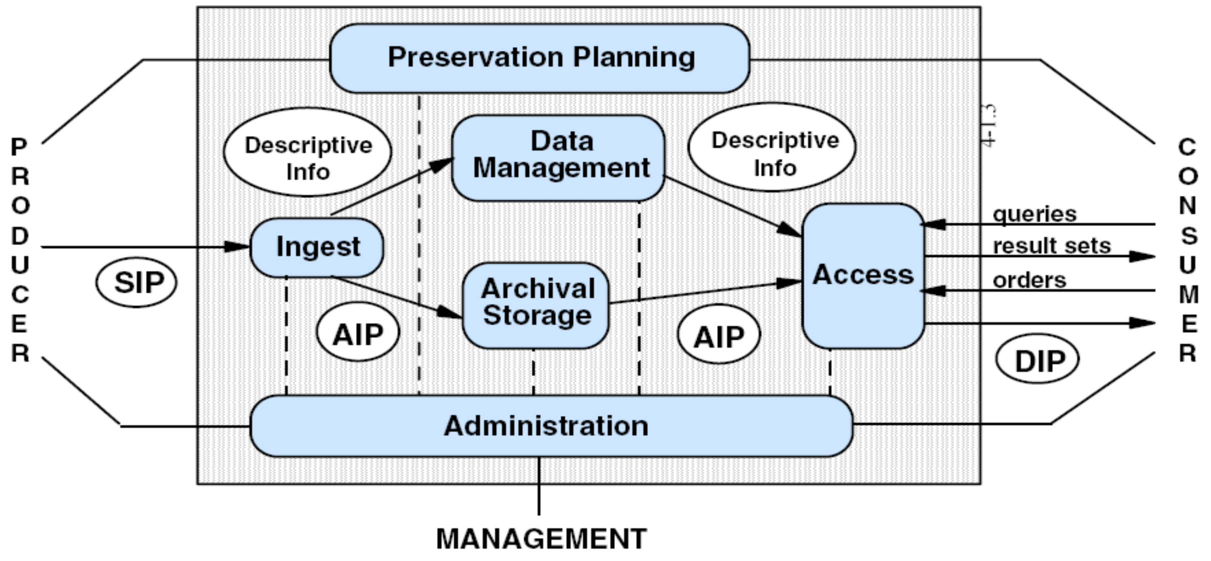
REFERENCE MODEL FOR AN OPEN ARCHIVAL INFORMATION SYSTEM (OAIS)

DRAFT RECOMMENDED STANDARD

CCSDS 650.0-P-1.1

PINK BOOK
August 2009


Producer
Submission Information Packages
OAIS
Archival Information Packages
Legend
= Entity
= Information Package Data Object
= Data Flow
queries
result sets
orders
Dissemination Information Packages
Consumer



Climate Data Services API

CDS Client Stack

CDS Command Line Interpreter

- The CDS WS Client and CDS Library provide the foundation upon which we are building a CDS Command Line Interpreter (CLI) that supports interactive sessions and processes Python scripts.

CDS Library

- The CDS Library is a Python-based library that sits atop the CDS WS Client and exposes the OAIS-inspired CDS Reference Model abstractions to calling applications.
- The CDS Library contains methods that support the basic primitives (ingest, query, order, etc.) as well as extended utilities that combine these primitives into automated multi-step canonical ops (avg, max, min, etc.).
- Extended utilities provide a place for community contributions ("adaptive construction" – a way we can grow analytic capabilities and engage the community to help with Big Data challenges a la GalaxyZoo ...)

CDS Web Services Client

- The CDS Web Services Client is a RESTful service that encapsulates inbound and outbound interactions with various climate data services.

CDS CLI

```
Welcome to the NASA GSFC CISTO Climate Data Services (cds).
Type help or ? to list commands.

(nasa-gsfc-cisto-cds) order MAS parms!
GetAverageByVariable_TimeRange_SpatialExtent_VerticalExtent
&operation=avg&variable_list=T&start_date=201101&end_date=201102&avg_peri
od=2&min_lon=-125&min_lat=24&max_lon=-66&max_lat=50&start_level=13&end
_level=13'

(nasa-gsfc-cisto-cds) execute HADOOP mapreduce jar!/opt/cds/bin/cds-mas-
mapreduce.jar  inputPath!/opt/cds/seq-input/merra/2011 outputPath!/opt/cds/
merra_2011_mr_seqout/npana
```

CDS Library

```
Class CDSLibrary(object):

   def  order(self, service, parms):
           cds_ws.order(service, parms)

   def  avg(self, service, parms, destination):
           sessionId = cds_ws.order(service, parms)
           response = cds_ws.status(service, sessionId)
         ……. Loop until result is available
           cds_ws.download(service, sessionId, destination)
```

CDS WS Client

```
Class CDSWSClient(object):
        def  order(self, service, parms):
              # validate/format parms
                  u = urllib.urlopen(url + service + 'order.php?' + parms
                  data = u.read()
               return data
        def  status(self,  service, sessionId):
                  u = urllib.urlopen(url + service + 'status.php?' + sessionId
                  data = u.read()
               return data
        def  download(self, service, sessionId):
                  u = urllib.urlopen(url + service + 'download.php?' + sessionId
              data = u.readlines
              u.close()
               ……
```

Climate Data Services API

CDS Client Stack

CDS Applications

- More complex applications can use methods imported from the Library.
- These full Python programs that can provide procedural controls, multiprocessing, etc.

CDS Scripts

- Scripts can call on the CDS Library, providing a simple way of automating synchronous interactions with MERRA/AS.

Note - These scripts and programs could be folded back into the library for general distribution if they are of value to the broader community ...

CDS Applications

```
[gtamkin@localhost python]$ more ./user_app_ext.py
from cds import CDSApi
cds_api = CDSApi()

service = 'MAS'
north_american_parms =
'GetAverageByVariable_TimeRange_SpatialExtent_VerticalExtent
&operation=avg&variable_list=T&start_date=201101&end_date=201102&avg_per
iod=2&min_lon=-125&min_lat=24&max_lon=-66&max_lat=50&start_level=13&en
d_level=13'
destination=/home/gtamkin/avg-out'

Class UserAppExt(object):

if __name__ == '__main__':
    sessionId = cds_api.avg(service, north_american_parms, destination)
    print "processing complete for = " + filename
```

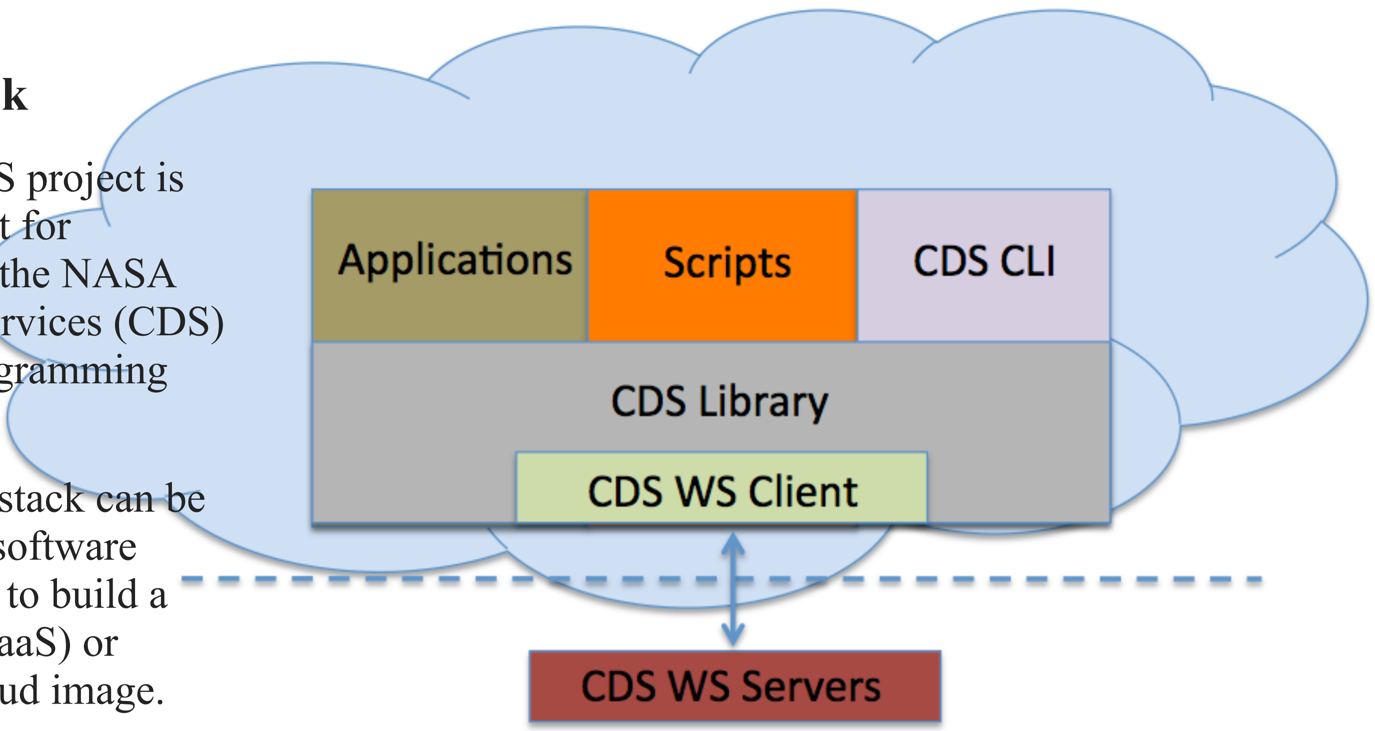
CDS Scripts

```
#!/usr/bin/env python
import time

from CDSLibrary import CDSApi
from wei_input import WEIInput
wei_exp = WEIInput()

# The rest of the file is run by the Python interpreter.
__doc__ = """This string is treated as the module docstring."""

service = wei_exp.getService()
catalog = wei_exp.getInput()
destination = wei_exp.getDestination()

cds_lib = CDSApi()
logger = cds_lib.getLogger()

start_time = time.time()

logger.debug("Generating: ca_avg_temp")
input = cds_lib.encode(catalog["ca_avg_temp_dictionary"])
cds_lib.avg(service, input, destination)

exit()
```

Climate Data Services API

CDS Client Stack

- The MERRA/AS project is the starting point for development of the NASA Climate Data Services (CDS) Application Programming Interface (API).
- The CDS client stack can be distributed as a software package or used to build a cloud service (SaaS) or distributable cloud image.
- This approach to API design focuses on the specific analytic requirements of the climate sciences — and moves us toward a notion of ***Climate Analytics-as-a-Service (CAaaS) …***



CDS Applications

```
[gtamkin@localhost python]$ more ./user_app_ext.py
from cds import CDSApi
cds_api = CDSApi()

service = 'MAS'
north_american_parms =
'GetAverageByVariable_TimeRange_SpatialExtent_VerticalExtent
&operation=avg&variable_list=T&start_date=201101&end_date=201102&a
vg_period=2&min_lon=-125&min_lat=24&max_lon=-66&max_lat=50&start
_level=13&end_level=13'
destination=/home/gtamkin/avg-out'

Class UserAppExt(object):

if __name__ == '__main__':
    sessionId = cds_api.avg(service, north_american_parms, destination)
    print "processing complete for = " + filename
```

CDS Scripts

```
#!/usr/bin/env python
import time

from CDSLibrary import CDSApi
from wei_input import WEIInput
wei_exp = WEIInput()

# The rest of the file is run by the Python interpreter.
__doc__ = """This string is treated as the module docstring."""

service = wei_exp.getService()
catalog = wei_exp.getInput()
destination = wei_exp.getDestination()

cds_lib = CDSApi()
logger = cds_lib.getLogger()

start_time = time.time()

logger.debug("Generating: ca_avg_temp")
input = cds_lib.encode(catalog["ca_avg_temp_dictionary"])
cds_lib.avg(service, input, destination)

exit()
```

CDS Library

```
Class CDSLibrary(object):

    def  order(self, service, parms):
            cds_ws.order(service, parms)

    def  avg(self, service, parms, destination):
            sessionId = cds_ws.order(service, parms)
            response = cds_ws.status(service, sessionId)
         ……. Loop until result is available
            cds_ws.download(service, sessionId, destination)
```

CDS Reference Model

Ingest – Submit/register a Submission Information Package (SIP).

Query – Retrieve data from a pre-determined service request (synchronous).

Order – Request data from a pre-determined service request (asynchronous).

Download – Retrieve a Dissemination Information Package (DIP).

Status – Track progress of service activity.

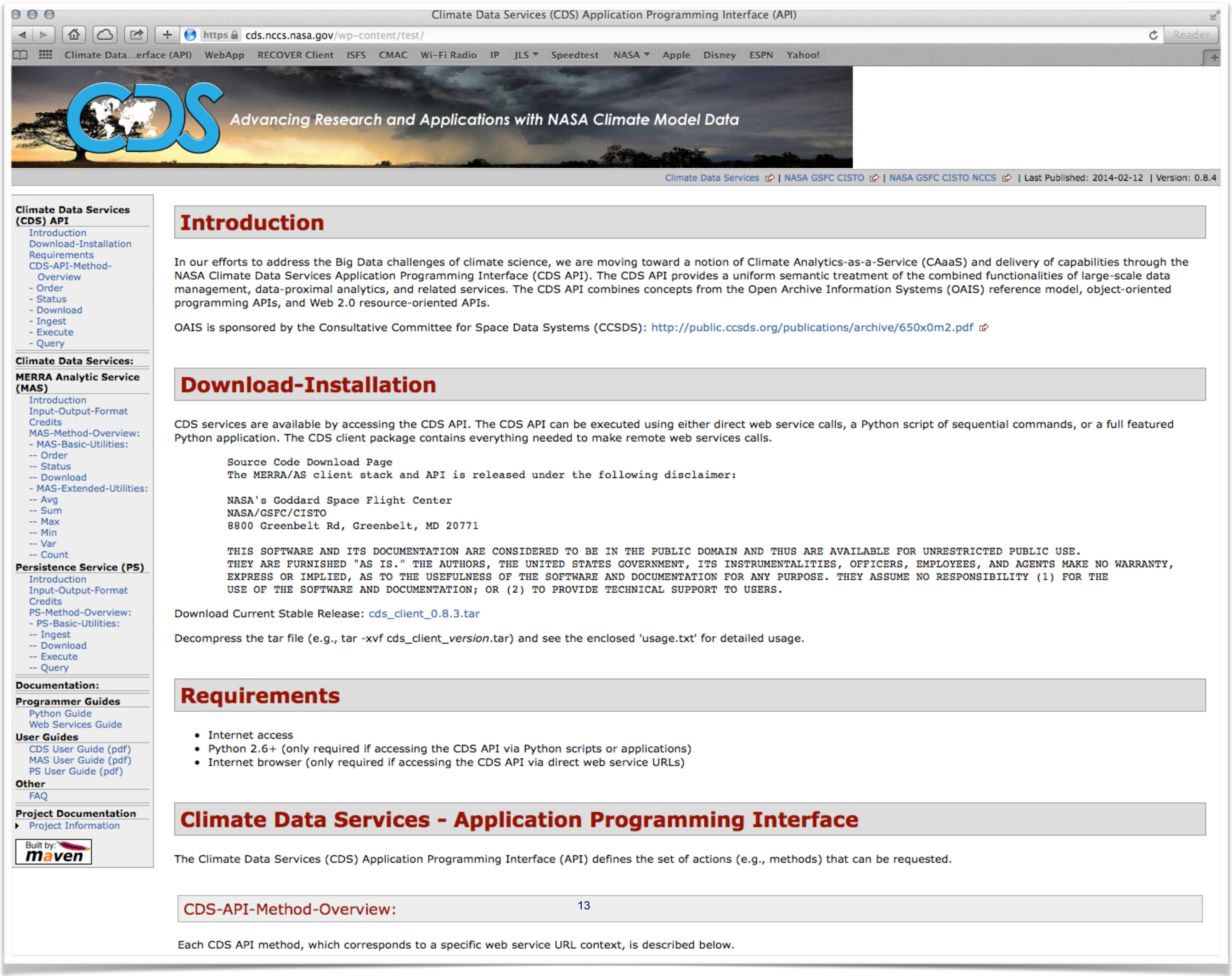
Execute – Initiate a service-definable extension. Allows for parameterized growth without API change.

CDS WS Client

```
Class CDSWSClient(object):
        def  order(self, service, parms):
               # validate/format parms
                    u = urllib.urlopen(url + service + 'order.php?' + parms
                    data = u.read()
                return data
        def  status(self,  service, sessionId):
                    u = urllib.urlopen(url + service + 'status.php?' + sessionId
                    data = u.read()
                return data
        def  download(self, service, sessionId):
                    u = urllib.urlopen(url + service + 'download.php?' + sessionId
               data = u.readlines
               u.close()
                ......
```

CDS CLI

```
Welcome to the NASA GSFC CISTO Climate Data Services (cds).
Type help or ? to list commands.

(nasa-gsfc-cisto-cds) order MAS parms!
GetAverageByVariable_TimeRange_SpatialExtent_VerticalExtent
&operation=avg&variable_list=T&start_date=201101&end_date=201102&a
vg_period=2&min_lon=-125&min_lat=24&max_lon=-66&max_lat=50&start
_level=13&end_level=13'

(nasa-gsfc-cisto-cds) execute HADOOP mapreduce jar!/opt/cds/bin/cds-
mas-mapreduce.jar  inputPath!/opt/cds/seq-input/merra/2011 outputPath!/
opt/cds/merra_2011_mr_seqout/npana
```


Climate Data Services (CDS) Application Programming Interface (API)
https cds.nccs.nasa.gov/wp-content/test/
Reader
Climate Data...erface (API) WebApp RECOVER Client ISFS CMAC Wi-Fi Radio IP JLS Speedtest NASA Apple Disney ESPN Yahoo!
CDS
Advancing Research and Applications with NASA Climate Model Data

Climate Data Services | NASA GSFC CISTO | NASA GSFC CISTO NCCS | Last Published: 2014-02-12 | Version: 0.8.4

Climate Data Services (CDS) API
- Introduction
- Download-Installation
- Requirements
- CDS-API-Method-Overview
- - Order
- - Status
- - Download
- - Ingest
- - Execute
- - Query

Climate Data Services:

MERRA Analytic Service (MAS)
- Introduction
- Input-Output-Format
- Credits
- MAS-Method-Overview:
- - MAS-Basic-Utilities:
- -- Order
- -- Status
- -- Download
- - MAS-Extended-Utilities:
- -- Avg
- -- Sum
- -- Max
- -- Min
- -- Var
- -- Count

Persistence Service (PS)
- Introduction
- Input-Output-Format
- Credits
- PS-Method-Overview:
- - PS-Basic-Utilities:
- -- Ingest
- -- Download
- -- Execute
- -- Query

Documentation:

Programmer Guides
- Python Guide
- Web Services Guide

User Guides
- CDS User Guide (pdf)
- MAS User Guide (pdf)
- PS User Guide (pdf)

Other
- FAQ

Project Documentation
- Project Information

Built by: maven

Introduction

In our efforts to address the Big Data challenges of climate science, we are moving toward a notion of Climate Analytics-as-a-Service (CAaaS) and delivery of capabilities through the NASA Climate Data Services Application Programming Interface (CDS API). The CDS API provides a uniform semantic treatment of the combined functionalities of large-scale data management, data-proximal analytics, and related services. The CDS API combines concepts from the Open Archive Information Systems (OAIS) reference model, object-oriented programming APIs, and Web 2.0 resource-oriented APIs.

OAIS is sponsored by the Consultative Committee for Space Data Systems (CCSDS): http://public.ccsds.org/publications/archive/650x0m2.pdf

Download-Installation

CDS services are available by accessing the CDS API. The CDS API can be executed using either direct web service calls, a Python script of sequential commands, or a full featured Python application. The CDS client package contains everything needed to make remote web services calls.

```
Source Code Download Page
The MERRA/AS client stack and API is released under the following disclaimer:

NASA's Goddard Space Flight Center
NASA/GSFC/CISTO
8800 Greenbelt Rd, Greenbelt, MD 20771

THIS SOFTWARE AND ITS DOCUMENTATION ARE CONSIDERED TO BE IN THE PUBLIC DOMAIN AND THUS ARE AVAILABLE FOR UNRESTRICTED PUBLIC USE.
THEY ARE FURNISHED "AS IS." THE AUTHORS, THE UNITED STATES GOVERNMENT, ITS INSTRUMENTALITIES, OFFICERS, EMPLOYEES, AND AGENTS MAKE NO WARRANTY,
EXPRESS OR IMPLIED, AS TO THE USEFULNESS OF THE SOFTWARE AND DOCUMENTATION FOR ANY PURPOSE. THEY ASSUME NO RESPONSIBILITY (1) FOR THE
USE OF THE SOFTWARE AND DOCUMENTATION; OR (2) TO PROVIDE TECHNICAL SUPPORT TO USERS.
```

Download Current Stable Release: cds_client_0.8.3.tar

Decompress the tar file (e.g., tar -xvf cds_client_*version*.tar) and see the enclosed 'usage.txt' for detailed usage.

Requirements

- Internet access
- Python 2.6+ (only required if accessing the CDS API via Python scripts or applications)
- Internet browser (only required if accessing the CDS API via direct web service URLs)

Climate Data Services - Application Programming Interface

The Climate Data Services (CDS) Application Programming Interface (API) defines the set of actions (e.g., methods) that can be requested.

CDS-API-Method-Overview:

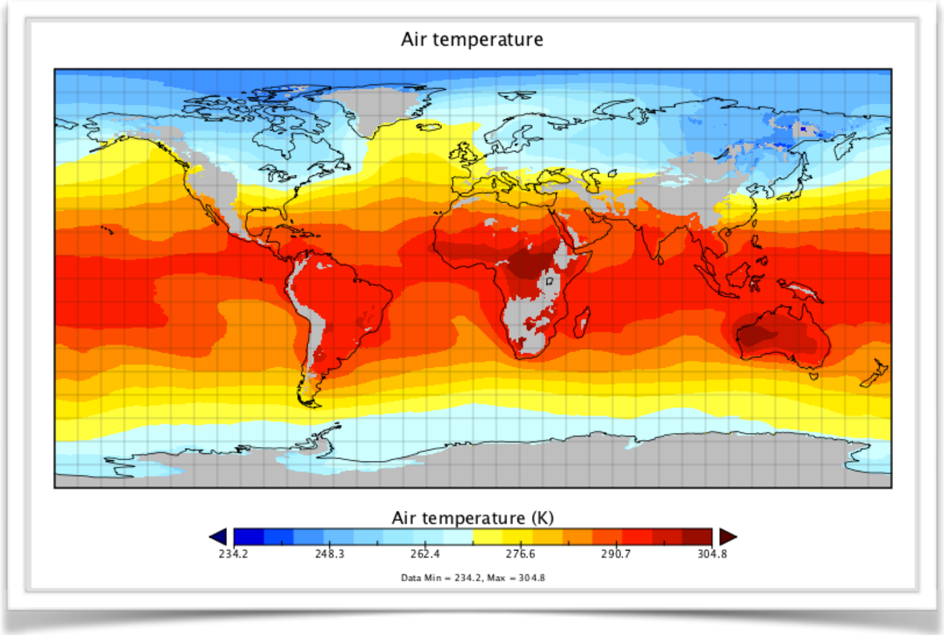
Each CDS API method, which corresponds to a specific web service URL context, is described below.

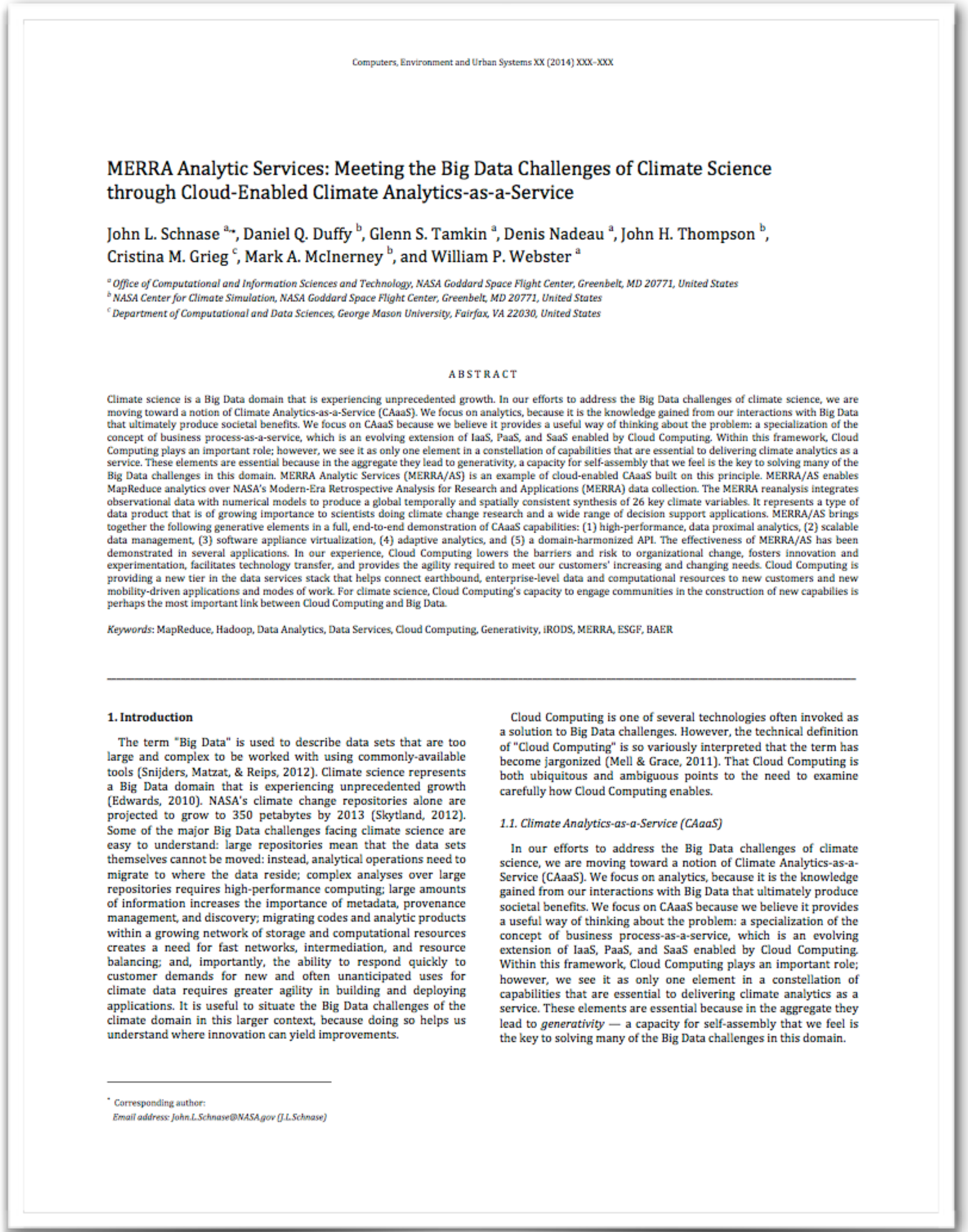
MERRA Analytic Services (MERRA/AS)

John Schnase, Glenn Tamkin,
Dan Duffy, and Mark McInerney

NASA Goddard Space Flight Center

March 20, 2014




Computers, Environment and Urban Systems XX (2014) XXX–XXX

MERRA Analytic Services: Meeting the Big Data Challenges of Climate Science through Cloud-Enabled Climate Analytics-as-a-Service

John L. Schnase [a,*], Daniel Q. Duffy [b], Glenn S. Tamkin [a], Denis Nadeau [a], John H. Thompson [b], Cristina M. Grieg [c], Mark A. McInerney [b], and William P. Webster [a]

[a] *Office of Computational and Information Sciences and Technology, NASA Goddard Space Flight Center, Greenbelt, MD 20771, United States*
[b] *NASA Center for Climate Simulation, NASA Goddard Space Flight Center, Greenbelt, MD 20771, United States*
[c] *Department of Computational and Data Sciences, George Mason University, Fairfax, VA 22030, United States*

ABSTRACT

Climate science is a Big Data domain that is experiencing unprecedented growth. In our efforts to address the Big Data challenges of climate science, we are moving toward a notion of Climate Analytics-as-a-Service (CAaaS). We focus on analytics, because it is the knowledge gained from our interactions with Big Data that ultimately produce societal benefits. We focus on CAaaS because we believe it provides a useful way of thinking about the problem: a specialization of the concept of business process-as-a-service, which is an evolving extension of IaaS, PaaS, and SaaS enabled by Cloud Computing. Within this framework, Cloud Computing plays an important role; however, we see it as only one element in a constellation of capabilities that are essential to delivering climate analytics as a service. These elements are essential because in the aggregate they lead to generativity, a capacity for self-assembly that we feel is the key to solving many of the Big Data challenges in this domain. MERRA Analytic Services (MERRA/AS) is an example of cloud-enabled CAaaS built on this principle. MERRA/AS enables MapReduce analytics over NASA's Modern-Era Retrospective Analysis for Research and Applications (MERRA) data collection. The MERRA reanalysis integrates observational data with numerical models to produce a global temporally and spatially consistent synthesis of 26 key climate variables. It represents a type of data product that is of growing importance to scientists doing climate change research and a wide range of decision support applications. MERRA/AS brings together the following generative elements in a full, end-to-end demonstration of CAaaS capabilities: (1) high-performance, data proximal analytics, (2) scalable data management, (3) software appliance virtualization, (4) adaptive analytics, and (5) a domain-harmonized API. The effectiveness of MERRA/AS has been demonstrated in several applications. In our experience, Cloud Computing lowers the barriers and risk to organizational change, fosters innovation and experimentation, facilitates technology transfer, and provides the agility required to meet our customers' increasing and changing needs. Cloud Computing is providing a new tier in the data services stack that helps connect earthbound, enterprise-level data and computational resources to new customers and new mobility-driven applications and modes of work. For climate science, Cloud Computing's capacity to engage communities in the construction of new capabilies is perhaps the most important link between Cloud Computing and Big Data.

Keywords: MapReduce, Hadoop, Data Analytics, Data Services, Cloud Computing, Generativity, iRODS, MERRA, ESGF, BAER

1. Introduction

The term "Big Data" is used to describe data sets that are too large and complex to be worked with using commonly-available tools (Snijders, Matzat, & Reips, 2012). Climate science represents a Big Data domain that is experiencing unprecedented growth (Edwards, 2010). NASA's climate change repositories alone are projected to grow to 350 petabytes by 2013 (Skytland, 2012). Some of the major Big Data challenges facing climate science are easy to understand: large repositories mean that the data sets themselves cannot be moved: instead, analytical operations need to migrate to where the data reside; complex analyses over large repositories requires high-performance computing; large amounts of information increases the importance of metadata, provenance management, and discovery; migrating codes and analytic products within a growing network of storage and computational resources creates a need for fast networks, intermediation, and resource balancing; and, importantly, the ability to respond quickly to customer demands for new and often unanticipated uses for climate data requires greater agility in building and deploying applications. It is useful to situate the Big Data challenges of the climate domain in this larger context, because doing so helps us understand where innovation can yield improvements.

Cloud Computing is one of several technologies often invoked as a solution to Big Data challenges. However, the technical definition of "Cloud Computing" is so variously interpreted that the term has become jargonized (Mell & Grace, 2011). That Cloud Computing is both ubiquitous and ambiguous points to the need to examine carefully how Cloud Computing enables.

1.1. Climate Analytics-as-a-Service (CAaaS)

In our efforts to address the Big Data challenges of climate science, we are moving toward a notion of Climate Analytics-as-a-Service (CAaaS). We focus on analytics, because it is the knowledge gained from our interactions with Big Data that ultimately produce societal benefits. We focus on CAaaS because we believe it provides a useful way of thinking about the problem: a specialization of the concept of business process-as-a-service, which is an evolving extension of IaaS, PaaS, and SaaS enabled by Cloud Computing. Within this framework, Cloud Computing plays an important role; however, we see it as only one element in a constellation of capabilities that are essential to delivering climate analytics as a service. These elements are essential because in the aggregate they lead to *generativity* — a capacity for self-assembly that we feel is the key to solving many of the Big Data challenges in this domain.

* Corresponding author:
Email address: John.L.Schnase@NASA.gov (J.L.Schnase)